CMPT 215.3 IN-LAB EXAMINATION
(Group C)
November 2, 2000

Total Marks: 10   **OPEN BOOK and OPEN NOTES**
Time: 90 minutes (3:00pm-4:30pm)

Instructions

Both questions are programming questions. You'll need to design algorithms, write, debug, and test MIPS assembly programs. The marks for each question are as indicated. Allocate your time accordingly.

1. (**5 marks**) Read in from the keyboard a sequence of **non-zero** integers finished by a 0, and print the minimum and the maximum of them. Note that the last 0 does not belong to this sequence of integers. For example, if the user inputs the following four numbers: 10, 3, 7, 0, the output should be:

> *The minimum is 3*
> *The maximum is 10*

2. (**5 marks**) Implement function **int *insert*(int *index*, int *data*, int *array_addr*, int *array_len*)** that inserts an integer into an array of integers at the location specified by the *index*. Assume that the *index* starts at 0 and always exists, i.e. $0 \leq index <$ *array_len*. In the function, the *data* is the integer to be inserted, the a*rray_addr* is the starting address of the integer array, and the *array_len* is the total number of integers in the array. Before the new *data* is inserted into the array at the location *index*, all elements from *index* until the end of the array are shifted one location to the right. This function returns the new length of the array. For example, assume the original array is 7, 24, 15, 3, 9 (whose length is 5), after you insert 10 at index 2, the array is changed to 7, 24, **10**, 15, 3, 9, and the length of which becomes 6.

In your ***main***, first allocate a space for 8 words and initialize them to be *7, 24, 15, 3, 9, 0, 0, 0*, then test the function ***insert*** as follows:

> *array_len = insert*(2, 10, *array_addr*, *array_len*)
> *array_len = insert* (0, 13, *array_addr*, *array_len*)
> *array_len = insert* (*array_len* - 1, 11, *array_addr*, *array_len*)

Print all the numbers of the array after each of the three calls. Note that " " (space) should be used to separate array elements from each other when printing the array.